Source Code  
Section 9, Lecture 67

The downloadable file is in the resources folder. The source code itself, is down below:

/AddPhoto

1. import React, {Component} from 'react'
2. class AddPhoto extends Component {
3. constructor() {
4. super()
5. thisthis.handleSubmit = this.handleSubmit.bind(this)
6. }
7. handleSubmit(event) {
8. event.preventDefault();
9. const imageLink = event.target.elements.link.value
10. const description = event.target.elements.description.value
11. const post = {
12. id: Number(new Date()),
13. description: description,
14. imageLink: imageLink
15. }
16. if (description && imageLink){
17. this.props.startAddingPost(post)
18. this.props.onHistory.push('/')
19. }
20. }
21. render() {
22. return (
23. **<div>**
24. **<div** className="form"**>**
25. **<form** onSubmit={this.handleSubmit}**>**
26. **<input** type ="text" placeholder="Link" name="link"**/>**
27. **<input** type ="text" placeholder="Desciption" name="description"**/>**
28. **<button>** Post **</button>**
29. **</form>**
30. **</div>**
31. **</div>**
32. )
33. }
34. }
35. export default AddPhoto

**/App**

1. import Main from './Main'
2. import {connect} from 'react-redux'
3. import {bindActionCreators} from 'redux'
4. import \* as actions from '../redux/actions'
5. import {withRouter} from 'react-router'
6. function mapStateToProps(state) {
7. return {
8. posts: state.posts,
9. comments: state.comments
10. }
11. }
12. function mapDispatchToProps(dispatch) {
13. return bindActionCreators(actions, dispatch)
14. }
15. const App = withRouter(connect(mapStateToProps, mapDispatchToProps)(Main))
16. export default App

**/Comments**

1. import React, {Component} from 'react'
2. class Comments extends Component {
3. constructor() {
4. super()
5. thisthis.handleSubmit = this.handleSubmit.bind(this)
6. }
7. handleSubmit(event) {
8. event.preventDefault()
9. const comment = event.target.elements.comment.value
10. this.props.startAddingComment(comment, this.props.id)
11. event.target.elements.comment.value = ''
12. }
13. render() {
14. return **<div** className="comment"**>**
15. {
16. this.props.comments.map((comment, index) =**>** {
17. return (
18. **<p** key={index}**>** {comment} **</p>**
19. )
20. })
21. }
22. **<form** className="comment-form" onSubmit={this.handleSubmit}**>**
23. **<input** type="text" placeholder="comment" name="comment"**/>**
24. **<input** type="submit" hidden**/>**
25. **</form>**

28. **</div>**
29. }
30. }
31. export default Comments

**/Main**

1. import React, {Component} from 'react'
2. import PhotoWall from './PhotoWall'
3. import AddPhoto from './AddPhoto'
4. import {Route, Link} from 'react-router-dom'
5. import Single from './Single'
6. class Main extends Component {
8. state = { loading: true }
9. componentDidMount() {
10. this.props.startLoadingPost().then(() =**>** {
11. this.setState({loading: false})
12. })
13. this.props.startLoadingComments()
14. }
15. render() {
16. return (
18. **<div>**
19. **<h1>**
20. **<Link** to="/"**>** Photowall **</Link>**
21. **</h1>**
22. **<Route** exact path = "/" render={() =**>** (
23. **<div>**
24. **<PhotoWall** {...this.props} **/>**
25. **</div>**
26. )}**/>**
27. **<Route** path= "/AddPhoto" render = {({history}) =**>** (
28. **<AddPhoto** {...this.props} onHistory={history}**/>**
29. )}**/>**
30. **<Route** exact path="/single/:id" render = {(params) =**>** (
31. **<Single** loading={this.state.loading} {...this.props} {...params}**/>**
32. )}**/>**
34. **</div>**
35. )
36. }
37. }
38. export default Main

**/Photo**

1. import React from 'react';
2. import PropTypes from 'prop-types'
3. import {Link} from 'react-router-dom'
4. function Photo(props) {
5. const post = props.post
6. return **<figure** className="figure"**>**
7. **<Link** to={`/single/${post.id}`}**>** **<img** className= "photo" src={post.imageLink} alt={post.description}**/></Link>**
8. **<figcaption>** **<p>** {post.description} **</p>** **</figcaption>**
9. **<div** className = "button-container"**>**
10. **<button** onClick = {() =**>** {
11. props.startRemovingPost(props.index, post.id)
12. props.history.push('/')
13. }}**>** Remove **</button>**
14. **<Link** className="button" to={`/single/${post.id}`}**>**
15. **<div** className="comment-count"**>**
16. **<div** className="speech-bubble"**>** **</div>**
17. {props.comments[post.id] ? props.comments[post.id].length : 0 }
18. **</div>**
19. **</Link>**
20. **</div>**
21. **</figure>**
22. }
23. Photo.propTypes = {
24. post: PropTypes.object.isRequired,
25. }
26. export default Photo

/Photowall

1. import React from 'react'
2. import Photo from './Photo'
3. import PropTypes from 'prop-types'
4. import {Link} from 'react-router-dom'
5. //anchor tag, href attribute
6. function PhotoWall(props) {
7. return **<div>**
8. **<Link** className = "addIcon" to="/AddPhoto"**>** **</Link>**
9. **<div** className="photoGrid" **>**
10. {props.posts
11. .sort(function(x,y) {
12. return y.id - x.id
13. })
14. .map((post, index) =**>** **<Photo** key={index} post={post} {...props} index={index}**/>**)}
15. **</div>**
16. **</div>**
17. }
18. PhotoWall.propTypes = {
19. posts: PropTypes.array.isRequired,
20. }
21. export default PhotoWall

/single

1. import React, {Component} from 'react'
2. import Photo from './Photo'
3. import Comments from './Comments'
4. class Single extends Component {
5. render() {
6. const {match, posts} = this.props
7. const id = Number(match.params.id)
8. const post = posts.find((post) =**>** post.id === id)
9. const comments = this.props.comments[match.params.id] || []
10. const index = this.props.posts.findIndex((post) =**>** post.id === id)
11. if (this.props.loading === true) {
12. return **<div** className="loader"**>** ...loading **</div>**
13. } else if (post) {
14. return **<div** className='single-photo'**>**
15. **<Photo** post={post} {...this.props} index={index}**/>**
16. **<Comments** startAddingComment={this.props.startAddingComment} comments={comments} id = {id}**/>**
17. **</div>**
18. } else {
19. return **<h1>** ...no post found **</h1>**
20. }
22. }
23. }
24. export default Single

**/Title**

1. import React from 'react'
2. function Title(props) {
3. return **<h1>** {props.title} **</h1>**
5. }
6. export default Title

**/Posts**

1. const posts = []
2. export default posts

**/Config**

1. import \* as firebase from 'firebase'
2. var config = {
3. apiKey: "AIzaSyAWZObGQDfGbplDsJ0QXZGfZ22Ph9vlkgc",
4. authDomain: "photowall-45333.firebaseapp.com",
5. databaseURL: "https://photowall-45333.firebaseio.com",
6. projectId: "photowall-45333",
7. storageBucket: "photowall-45333.appspot.com",
8. messagingSenderId: "586411564955"
9. }
11. firebase.initializeApp(config)
12. const database = firebase.database()
13. export {database}

**/actions**

1. import {database} from '../database/config'
2. export function startAddingPost(post) {
3. return (dispatch) =**>** {
4. return database.ref('posts').update({[post.id]: post}).then(() =**>** {
5. dispatch(addPost(post))
6. }).catch((error) =**>** {
7. console.log(error)
8. })
9. }
10. }
11. export function startLoadingPost() {
12. return (dispatch) =**>** {
13. return database.ref('posts').once('value').then((snapshot) =**>** {
14. let posts = []
15. snapshot.forEach((childSnapshot) =**>** {
16. posts.push(childSnapshot.val())
17. })
18. dispatch(loadPosts(posts))
19. }).catch((error) =**>** {
20. console.log(error)
21. })
22. }
23. }
24. export function startRemovingPost(index, id) {
25. return (dispatch) =**>** {
26. return database.ref(`posts/${id}`).remove().then(() =**>** {
27. dispatch(removePost(index))
28. }).catch((error) =**>** {
29. console.log(error)
30. })
31. }
32. }
33. export function startAddingComment(comment, postId) {
34. return (dispatch) =**>** {
35. return database.ref('comments/'+postId).push(comment).then(() =**>** {
36. dispatch(addComment(comment, postId))
37. }).catch((error) =**>** {
38. console.log(error)
39. })
40. }
41. }
42. export function startLoadingComments() {
43. return (dispatch) =**>** {
44. return database.ref('comments').once('value').then((snapshot) =**>** {
45. let comments = {}
46. snapshot.forEach((childSnapshot) =**>** {
47. comments[childSnapshot.key] = Object.values(childSnapshot.val())
48. })
49. dispatch(loadComments(comments))
50. })
51. }
52. }
53. export function loadComments(comments) {
54. return {
55. type: 'LOAD\_COMMENTS',
56. comments
57. }
58. }
59. export function removePost(index) {
60. return {
61. type: 'REMOVE\_POST',
62. index
63. }
64. }
65. export function addPost(post) {
66. return {
67. type: 'ADD\_POST',
68. post
69. }
70. }
71. export function addComment(comment, postId) {
72. return {
73. type: 'ADD\_COMMENT',
74. comment,
75. postId
76. }
77. }
78. export function loadPosts(posts) {
79. return {
80. type: 'LOAD\_POSTS',
81. posts
82. }
83. }

**/reducer**

1. import \_posts from '../data/posts'
2. import {combineReducers} from 'redux'
3. function comments(state={}, action) {
4. switch (action.type) {
5. case 'ADD\_COMMENT':
7. if (!state[action.postId]) {
8. return {...state, [action.postId]: [action.comment]}
9. } else {
10. return {...state, [action.postId]: [...state[action.postId], action.comment] }
11. }
12. case 'LOAD\_COMMENTS': return action.comments
14. default: return state
15. }
16. return state
17. }
18. function posts(state = \_posts, action) {
19. switch (action.type) {
20. case 'REMOVE\_POST': return [...state.slice(0, action.index), ...state.slice(action.index + 1)]
21. case 'ADD\_POST': return [...state, action.post]
22. case 'LOAD\_POSTS': return action.posts
23. default: return state
24. }
25. }
26. const rootReducer = combineReducers({posts, comments})
27. export default rootReducer

**/stylesheet**

1. html {
2. font-size: 10px;
3. font-family: sans-serif;
4. }
5. p {
6. font-size: 1.6rem;
7. line-height: 1.5;
8. }
9. h1 {
10. font-family: billabong, 'billabongregular';
11. text-align: center;
12. font-weight: 100;
13. font-size: 13rem;
14. margin: 2rem 0;
15. letter-spacing: -1px;
16. color: black;
17. text-decoration: none;
18. }
19. h1 a {
20. color: black;
21. text-decoration: none;
22. }
23. h1 a:focus {
24. outline: 0;
25. }
26. @font-face {
27. font-family: 'billabongregular';
28. src: url("https://cdn.rawgit.com/milktronics/beaglegr.am/master/public/fonts/billabong-webfont.eot");
29. src: url("https://cdn.rawgit.com/milktronics/beaglegr.am/master/public/fonts/billabong-webfont.eot?#iefix") format('embedded-opentype'), url("https://cdn.rawgit.com/milktronics/beaglegr.am/master/public/fonts/billabong-webfont.woff") format('woff'), url("https://cdn.rawgit.com/milktronics/beaglegr.am/master/public/fonts/billabong-webfont.ttf") format('truetype'), url("https://cdn.rawgit.com/milktronics/beaglegr.am/master/public/fonts/billabong-webfont.svg#billabongregular") format('svg');
30. font-weight: normal;
31. font-style: normal;
32. }
33. body {
34. background: offwhite;
35. }
36. .photoGrid {
37. display: flex;
38. flex-wrap: wrap;
39. max-width: 1000px;
40. margin: 0 auto;
41. }
42. .figure {
43. flex-basis: calc(33.333% - 4rem);
44. border: 1px solid #d3d3d3;
45. padding: 2rem;
46. flex-grow: 1;
47. margin: 0 2rem 2rem 2rem;
48. }
49. .figure .single-photo {
50. flex-basis: calc(33.333% - 4rem);
51. flex-grow: 1;
52. flex-shrink: 0;
53. margin: 0 2rem 2rem 2rem;
54. padding: 2rem;
55. border: 1px solid #d3d3d3;
56. background: #fff;
57. position: relative;
58. }
59. .single-photo {
60. max-width: 1000px;
61. margin: 0 auto;
62. display: flex;
63. background: #fff;
64. border: black;
65. }
66. .single-photo .figure {
67. box-shadow: none;
68. margin: 0 2rem 0 0;
69. border: 0;
70. padding: 0;
71. flex: 1 0 60%;
72. max-width: 60%;
73. }
74. .single-photo .comments {
75. flex: 1 0 40%;
76. max-width: 40%;
77. }
78. .single-photo .photo {
79. width: 100%;
80. margin: 0;
81. }
82. .photo {
83. width: calc(100% + 4rem);
84. margin-left: -2rem;
85. margin-top: -2rem;
86. max-width: none;
87. }
88. .comment{
89. width: 100%
90. }
91. .comment p {
92. border-bottom: 1px solid #d3d3d3;
93. padding: 0.5rem 0;
94. }
95. .comment-form input,
96. .comment-form textarea {
97. width: 100%;
98. border: 0;
99. font-size: 1.3rem;
100. padding: 1rem 0;
101. border-bottom: 1px solid #d3d3d3;
102. outline: none;
103. resize: vertical;
104. }
105. .addIcon {
106. background: url("https://image.flaticon.com/icons/svg/60/60740.svg") center no-repeat;
107. background-size: contain;
108. padding: 40px;
109. display: table;
110. margin: 20px auto;
111. border: none;
112. }
113. .button-container {
114. margin: 0 auto;
115. display: flex;
116. align-items: center;
117. justify-content: center;
118. }
119. button, .button {
120. border: 2px solid #f2f2f2;
121. background: none;
122. flex-basis: 48%;
123. display: inline-block;
124. line-height: 2;
125. text-decoration: none;
126. padding: 5px;
127. text-align: center;
128. font-size: 15px;
129. color: red;
130. transition: all 0.2s;
131. box-sizing: padding-box;
133. }
134. button:hover,.button:hover, button:focus, .button:focus {
135. border-color: #00f;
136. outline: 0;
137. }
138. .speech-bubble {
139. background: url("https://d30y9cdsu7xlg0.cloudfront.net/png/51904-200.png") center no-repeat;
140. background-size: contain;
141. padding: 10px;
142. margin: 5px;
143. display: inline-block;
144. border: none;
145. position: relative;
146. }
147. .comment-count {
148. margin: 0 auto;
149. display: flex;
150. align-items: center;
151. justify-content: center;
152. color: black
153. }
154. .form {
155. width: 360px;
156. padding: 0 0 0;
157. margin: auto;
158. }
160. .form input {
161. font-family: sans-serif;
162. outline: 0;
163. background: #f2f2f2;
164. width: 100%;
165. border: 0;
166. margin: 0 0 15px;
167. padding: 15px;
168. box-sizing: border-box;
169. font-size: 14px;
170. }
171. .form button {
172. font-family: "Roboto", sans-serif;
173. text-transform: uppercase;
174. outline: 0;
175. background:#065784;
176. width: 100%;
177. border: 0;
178. padding: 15px;
179. color: #FFFFFF;
180. font-size: 14px;
181. -webkit-transition: all 0.3 ease;
182. transition: all 0.3 ease;
183. cursor: pointer;
184. }
185. .form button:hover,.form button:active,.form button:focus {
186. background: #7B8589;
187. }
188. .loader {
189. height: 20%;
190. width: 100%;
191. display: flex;
192. position: fixed;
193. align-items: center;
194. justify-content: center;
195. font-size: 28pt;
196. }

**/index**

1. import React from 'react'
2. import ReactDOM from 'react-dom'
3. import './styles/stylesheet.css'
4. import {BrowserRouter} from 'react-router-dom'
5. import {createStore, applyMiddleware} from 'redux'
6. import rootReducer from './redux/reducer'
7. import {Provider} from 'react-redux'
8. import App from './Components/App'
9. import thunk from 'redux-thunk'
10. import {database} from './database/config'
11. const store = createStore(rootReducer, window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_ && window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_(), applyMiddleware(thunk))
12. ReactDOM.render(**<Provider** store={store}**><BrowserRouter><App/></BrowserRouter></Provider>**, document.getElementById('root'))

**/html**

1. <!DOCTYPE html**>**
2. **<html** lang="en"**>**
3. **<head>**
4. **<meta** charset="utf-8"**>**
5. **<meta** name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no"**>**
6. **<meta** name="theme-color" content="#000000"**>**
7. <!--
8. manifest.json provides metadata used when your web app is added to the
9. homescreen on Android. See https://developers.google.com/web/fundamentals/engage-and-retain/web-app-manifest/
10. --**>**
11. **<link** rel="manifest" href="%PUBLIC\_URL%/manifest.json"**>**
12. **<link** rel="shortcut icon" href="%PUBLIC\_URL%/favicon.ico"**>**
13. <!--
14. Notice the use of %PUBLIC\_URL% in the tags above.
15. It will be replaced with the URL of the `public` folder during the build.
16. Only files inside the `public` folder can be referenced from the HTML.
17. Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC\_URL%/favicon.ico" will
18. work correctly both with client-side routing and a non-root public URL.
19. Learn how to configure a non-root public URL by running `npm run build`.
20. --**>**
21. **<title>**React App**</title>**
22. **</head>**
23. **<body>**
24. **<noscript>**
25. You need to enable JavaScript to run this app.
26. **</noscript>**
27. **<div** id="root"**></div>**
28. <!--
29. This HTML file is a template.
30. If you open it directly in the browser, you will see an empty page.
31. You can add webfonts, meta tags, or analytics to this file.
32. The build step will place the bundled scripts into the **<body>** tag.
33. To begin the development, run `npm start` or `yarn start`.
34. To create a production bundle, use `npm run build` or `yarn build`.
35. --**>**
36. **</body>**
37. **</html>**